IN THE CLAIMS:

This listing of claims will replace all prior versions, and listing of claims, in the Application.

Listing of claims:

1 – 5. Canceled.

6. (Currently amended) A method of debugging a user defined flow of a program by executing an optimized flow derived from the user defined flow, each of said optimized flow and said user defined flow comprising a plurality of nodes connected by a plurality of connections, said method comprising:

constructing a stack associated with a terminal of a given node of said plurality of nodes in said optimized flow, said stack being tangibly embodied on a storage-type computer readable medium, said terminal connecting to a given optimized flow connection of said plurality of connections in said optimized flow, where said given optimized flow connection is associated with at least one user flow connection of said plurality of connections in said user defined flow;

reporting imminent execution of said given optimized flow connection;

receiving an instruction to push an indication of a particular user flow connection, among said at least one user flow connection associated with said given optimized flow connection, into said stack, the indication enabling the user to debug the user defined flow; and

responsive to receiving said instruction to push, push said indication of said particular user flow connection into said stack such that the user can debug the program code using the indication.

7. (Original) The method of claim 6 further comprising:

receiving an instruction to delay said execution of said given optimized flow connection; and

responsive to receiving said instruction to delay, delaying said execution of said given optimized flow connection pending receipt of a corresponding resume instruction.

8. (Original) The method of claim 6 further comprising:

receiving an instruction to resume said execution of said given optimized flow connection; and

responsive to receiving said instruction to resume, resuming said execution of said given optimized flow connection.

9. (Original) The method of claim 6 further comprising constructing a new stack associated with said terminal of said given node for each iteration of a loop.

10. (Currently amended) A runtime for executing an optimized flow that is derived from a user defined flow, the execution of the optimized flow being used by a user to debug the user defined flow, each of said optimized flow and said user defined flow comprising a plurality of nodes connected by a plurality of connections, said runtime operable to:

construct a stack associated with a terminal of a given node of said plurality of nodes in said optimized flow, said stack being tangibly embodied on a storage-type computer readable medium, said terminal connecting to a given

optimized flow connection of said plurality of connections in said optimized flow, where said given optimized flow connection is associated with at least one user flow connection of said plurality of connections in said user defined flow;

report imminent execution of said given optimized flow connection;

receive an instruction to push an indication of a particular user flow connection, among said at least one user flow connection associated with said given optimized flow connection, into said stack, the indication enabling the user to debug the user defined flow; and

push said indication of said particular user flow connection into said stack so that the user can debug the user defined flow.

11. (Currently amended) A computer readable medium containing computer-executable instructions which, when performed by a processor in a computer system for executing an optimized flow that is derived from a user defined flow, that is used to debug the user defined flow, each of said optimized flow and said user defined flow comprising a plurality of nodes connected by a plurality of connections, cause said computer system to:

construct a stack associated with a terminal of a given node of said plurality of nodes in said optimized flow, said stack being tangibly embodied on a storage-type computer readable medium, said terminal connecting to a given optimized flow connection of said plurality of connections in said optimized flow, where said given optimized flow connection is associated with at least one user flow connection of said plurality of connections in said user defined flow;

report imminent execution of said given optimized flow connection;

receive an instruction to push an indication of a particular user flow connection, among said at least one user flow connection associated with said given optimized flow connection, into said stack, the indication enabling the user to debug the user defined flow; and

push said indication of said particular user flow connection into said stack so that the user can debug the user defined flow.

12 - 16 Canceled.

17. (Currently amended) A debugger for debugging a user defined flow that has been compiled into an optimized flow, each of said optimized flow and said user defined flow comprising a plurality of nodes connected by a plurality of connections, said debugger operable to:

receive a report, from a runtime, of imminent execution of a given optimized flow connection of said plurality of connections in said optimized flow;

query said runtime to identify at least one user flow connection of said plurality of connections in said user defined flow associated with said given optimized flow connection;

determine whether a breakpoint has been placed on a first user flow connection of said at least one user flow connection in said user defined flow;

in response to determining whether a breakpoint has been placed, determine whether an indication of said first user flow connection exists in a stack associated with a terminal of a given node of said plurality of nodes in said optimized flow, said stack being tangibly embodied on a storage-type computer readable medium, said terminal connecting to said given optimized flow connection; and

in response to determining whether an indication of said first user flow connection, instruct said runtime to push an indication of said first user flow connection into said stack.

18. (Currently amended) A computer readable medium containing computer-executable instructions that, when performed by a processor in a computer system for debugging a user defined flow that has been compiled into an optimized flow, each of said optimized flow and said user defined flow comprising a plurality of nodes connected by a plurality of connections, cause said computer system to:

receive a report, from a runtime, of imminent execution of a given optimized flow connection of said plurality of connections in said optimized flow;

query said runtime to identify at least one user flow connection of said plurality of connections in said user defined flow associated with said given optimized flow connection;

determine whether a breakpoint has been placed on a first user flow connection of said at least one user flow connection in said user defined flow;

in response to determining whether a breakpoint has been placed, determine whether an indication of said first user flow connection exists in a stack associated with a terminal of a given node of said plurality of nodes in said optimized flow, said stack being tangibly embodied on a storage-type computer readable medium, said terminal connecting to said given optimized flow connection; and

in response to determining whether an indication of said first user flow connection, instruct said runtime to push an indication of said first user flow connection into said stack.

19. (Previously presented) A method of debugging a program code, the program code including at least one node having a breakpoint and a variable, the breakpoint for allowing a user to use a value of the variable to debug the program code, the method comprising:

generating an optimized flow from the program code, the generated optimized flow having at least two optimized nodes derived from the one node having the breakpoint;

executing the optimized code;

ensuring that two values of the variable are stored during execution of the optimized code, each one of the values being stored when one of the at least two optimized nodes is traversed; and

debugging the program code using the two stored values of the variable.

20. (Previously presented) The method of Claim 19 wherein the at least two optimized nodes are derived from the one node having the breakpoint when there is a loop in the program code that includes at least a first and second branches of code such that during a first iteration of the loop the first branch of code is traversed and during a second iteration of the loop the second branch of code is traversed and the node having the breakpoint is traversed at both the first and second iterations of the loop.

21. (Previously presented) A computer program product on a storage-type computer readable medium for allowing a user to debug a program code, the program code including at least one node having a breakpoint and a variable, the breakpoint for allowing a user to use a value of the variable to debug the program code, the computer program product comprising:

code means for generating an optimized flow from the program code, the generated optimized flow having at least two optimized nodes derived from the one node having the breakpoint;

code means for executing the optimized code;

code means for ensuring that two values of the variable are stored during execution of the optimized code, each one of the values being stored when one of the at least two optimized nodes is traversed thereby allowing the user to debug the program code using the stored values of the variable.

22. (Previously presented) The computer program product of Claim 21 wherein the at least two optimized nodes are derived from the one node having the breakpoint when there is a loop in the program code that includes at least a first and second branches of code such that during a first iteration of the loop

the first branch of code is traversed and during a second iteration of the loop the second branch of code is traversed and the node having the breakpoint is traversed at both the first and second iterations of the loop.

23. (Previously presented) A computer system being used to debug a program code, the program code including at least one node having a breakpoint and a variable, the breakpoint for allowing a user to use a value of the variable to debug the program code, the computer system comprising:

a storage device to store code data; and

a processor for processing the code data to generate an optimized flow from the program code, the generated optimized flow having at least two optimized nodes derived from the one node having the breakpoint, to execute the optimized code, to ensure that two values of the variable are stored during execution of the optimized code, each one of the values being stored when one of the at least two optimized nodes is traversed, the two stored values of the variable enabling the user to debug the program code.

24. (Previously presented) The computer system of Claim 23 wherein the at least two optimized nodes are derived from the one node having the breakpoint when there is a loop in the program code that includes at least a first and second branches of code such that during a first iteration of the loop the first branch of code is traversed and during a second iteration of the loop the second branch of code is traversed and the node having the breakpoint is traversed at both the first and second iterations of the loop.